Summary: Meal-Prep-Time Exercise

Problem Formulation: Two data sources are provided, **'orders.csv'** and **'restaurants.csv'**.
The objective of the exercise is to create a model that can estimate food preparation time for an order based on the sources provided.

Source 1: ***orders.csv***

contains information about specific **food orders**, including:

1. **order_acknowledged_at**: the date and time of day that the specified order is acknowledged.

2. **order_ready_at**: the date and time of day that the specified order is ready.

3. **order_value_gbp**: the value of the order in pounds.

4. **restaurant_id**: the id of the restaurant that received the order.

5. **number_of_items**: the number of items included in the order.

6. **prep_time_seconds**: the time taken to prepare the order in seconds (ready - acknowledged at).

Source 2: ***restaurants.csv***

contains information about specific **restaurants** including:

1. **restaurant_id**: the id of the specified restaurant.

2. **country**: the country of the restaurant.

3. **city** the city of the restaurant.

4. **type_of_food**: the type of cuisine that the restaurant specializes in.

Both sources contain variables that can be used for modelling the preparation time. We can join the two tables by the 'restaurant_id' since this column is common between both tables:

```
# merging 'orders' and 'restaurants' variables to single DataFrame
df = orders.merge(restaurants,on='restaurant_id') # merging by 'restaurant_id' column
```

this results in a new DataFrame containing additional information about each order including restaurant_id, restaurant cuisine, and the country and city of the restaurant. The following is a preview of the merged DataFrame structure based on a single row:

Index	order_acknowledged_at	order_ready_at	order_value_gbp	restaurant_id	number_of_items	prep_time_seconds	country	city	type_of_food
0	2015-06-01 12:28:28.952789+01:00	2015-06-01 14:12:09.474896+01:00	59.9	1326	2	6220	UK	London	burritos

Next, we need to consider features to use in the model. Since the **'order_acknowledged_at'** and **'order_ready_at'** columns are unformatted, we must convert them into a usable format. Once formatted, we are able to extract features from the columns including the day of week, hour, minute, or the second associated with each timestamp.

For the purpose of our model, we will only use features from the **'order_acknowledged_at'** column since we can only predict the preparation time at the point where the order is acknowledged, *not* the point at which the order has already been prepared.

In our notebook analysis, we calculated the median order preparation time based on the hour of the day that the order was acknowledged:

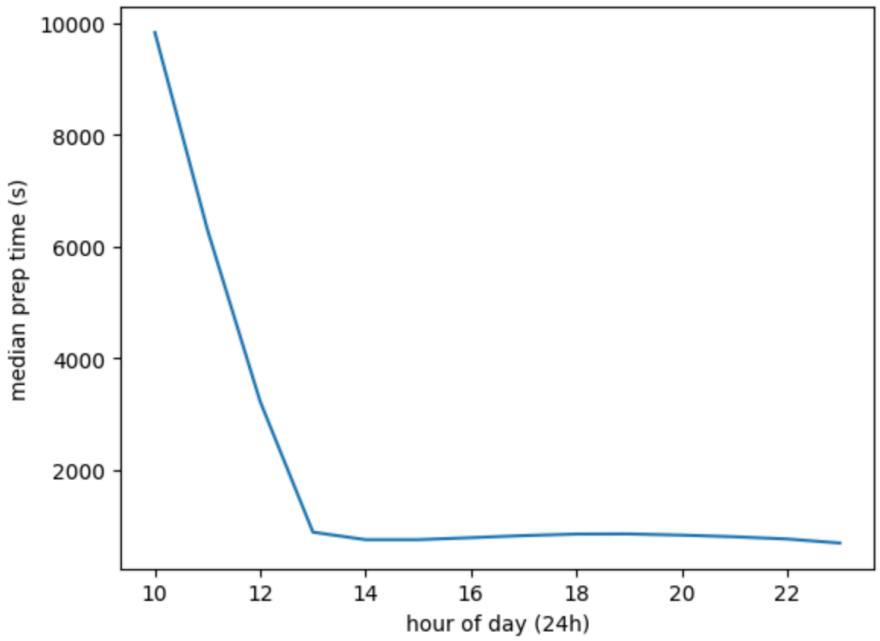

figure: *hour of day vs median preparation time*

Interestingly, the graph shows that the median preparation time for orders is significantly higher from 10AM to noon compared to afternoon hours. This suggests that the **'order_acknowledged_at'** time can significantly impact the preparation time for an order. The **minute of the day acknowledged** can be extracted from the timestamp using the following function:

```
df['minute_acknowledged'] = df['order_acknowledged_at'].apply(lambda x: x.hour*60 + x.minute) # minute acknowledged
```

as seen from the above code, the **minute_acknowledged** column takes the hour and minute of the **order_acknowledged_at** timestamp and combines these into a single feature.

Another part of the analysis was focused on variation within specified restaurants. To visualize this, the five restaurant_ids with the highest number of orders were determined and their respective medians and standard deviations were calculated:

	restaurant_id	median_prep_s	stdev_prep_s	median_prep_m	stdev_prep_m
0	408	945	1358	15.8	22.6
1	365	594	605	9.9	10.1
2	1939	835	1568	13.9	26.1
3	20	1067	1857	17.8	31.0
4	1390	513	1082	8.6	18.0

table: *prep time medians and standard deviations for sample restaurants*

We can see that that both median and standard deviation for order preparation time can vary widely from restaurant to restaurant. Therefore, features beyond restaurant characteristics may be needed to sufficiently predict order preparation time.

Experimental Model: Food Prep Time

We can model food preparation time as an output using the variables provided from 'orders.csv' and 'restaurants.csv'. However, before doing so we must construct our feature DataFrame and encode the 'country', 'city', and 'type_of_food' columns numerically:

Index	order _value_gbp	number _of_items	country	city	type_of_food	minute_acknowledged
0	59.9	2	3	14	13	748
1	24	8	3	14	13	1026
2	15.25	3	3	14	13	896
3	28.05	8	3	14	13	912
4	56.3	7	3	14	16	787

We will use the above six features to model a model for preparation time (order value in gbp, number of items, country, city, type of food and minute of day for the order acknowledged). Outliers for model_prep_time are removed before modelling. For the purpose of this summary, we will use the **XGBRegressor** algorithm to create a model for preparation time.

Based on our run of the model, the **mean absolute error** is calculated to be **480.1 seconds** or **8.0 minutes** when evaluated with test data.

The feature importances were also determined by the built in XGBRegressor function:

	importance
minute_acknowledged	0.376094
country	0.294947
city	0.091126
order_value_gbp	0.090822
type_of_food	0.081299
number_of_items	0.065712

-from this run of the model, **minute acknowledged** and **country** are attributed the highest feature importances in predicting order preparation time, followed by city, order value in gbp, type of food, and number of items.

Next Steps

The model would likely need more training data and an expanded feature set to be improved. Additional features of relevance should be determined if available.

Recommendations

The model may benefit from additional features outside the scope of the provided orders.csv or restaurants.csv files. For example, the type of items ordered are likely to impact preparation time. If a customer orders 20 soft drink cans, this would likely result in lower preparation time compared to an order of 20 food items, however both are treated as 20 items in the current version of the model which only specifies number of items rather than features pertaining to item types.